

Ankieta -
konsultacje Ministerstwa Cyfryzacji
dot. założeń konkursu na wdrożenie
open source



Stanowisko Stowarzyszenia QGIS Polska

Spis treści:

1. W jaki sposób dobrze zdefiniować open source na potrzeby konkursu? Czy definicja powinna odwoływać się do konkretnych licencji lub zapisów licencyjnych?	3
2. Jakie czynniki będą służyły skuteczności wdrożeń?	6
3. W jaki sposób zapewnić maksymalną replikowalność wdrażanych rozwiązań w innych jednostkach administracji publicznej?	9
4. Jakie czynniki powinny być brane pod uwagę przez podmioty realizujące pilotaże w zakresie bezpieczeństwa wdrażanych rozwiązań?	14
5. W jaki sposób najskuteczniej ułatwić pracownikom podmiotów realizujących pilotaże przejście z rozwiązań własnościowych na open source?	18
6. Jakie wsparcie techniczne i eksperckie powinno zostać zapewnione podmiotom realizującym pilotaże?	22
7. Pozostałe uwagi dotyczące planowanego konkursu	26

1. W jaki sposób dobrze zdefiniować open source na potrzeby konkursu? Czy definicja powinna odwoływać się do konkretnych licencji lub zapisów licencyjnych?

Rekomendujemy, aby na potrzeby konkursu open source zostało zdefiniowane precyzyjnie, dwuwarstwowo: po pierwsze przez prawa, jakie rozwiązanie daje administracji publicznej, a po drugie przez katalog akceptowalnych licencji oraz obowiązkowych cech wdrożeniowych. Sama deklaracja dostawcy, że rozwiązanie jest „otwarte”, „darmowe” albo „oparte na open source”, nie powinna być wystarczająca.

Wyniki badania MC-COI pokazują, że administracja publiczna działa dziś w środowisku silnie zdominowanym przez oprogramowanie własnościowe, szczególnie w obszarze systemów operacyjnych, pakietów biurowych, bezpieczeństwa, tożsamości, systemów finansowo-kadrowych czy obiegu dokumentów. Jednocześnie część jednostek posiada tylko częściowo aktualną ewidencję licencji oprogramowania. To oznacza, że konkurs powinien od początku ograniczać ryzyko niejasności licencyjnych i pozornej otwartości. Definicja open source musi być na tyle jednoznaczna, aby beneficjenci, wykonawcy, audytorzy i osoby odpowiedzialne za zamówienia publiczne rozumieli ją tak samo.

Proponujemy następującą definicję roboczą: za rozwiązanie open source na potrzeby konkursu należy uznać oprogramowanie, którego pełny kod źródłowy jest dostępny dla zamawiającego i społeczności, które może być używane w dowolnym celu, analizowane, kopiowane, uruchamiane, modyfikowane oraz dalej rozpowszechniane, w tym w wersjach zmodyfikowanych, na podstawie licencji spełniającej uznane kryteria wolnego i otwartego oprogramowania. Wymóg ten powinien obejmować nie tylko aplikację końcową, lecz także komponenty niezbędne do jej działania, dokumentację wdrożeniową, skrypty konfiguracyjne, interfejsy programistyczne, schematy wymiany danych oraz elementy konieczne do odtworzenia i utrzymania wdrożenia przez inny podmiot.

Definicja powinna wyraźnie rozróżniać open source od kilku kategorii, które nie powinny być uznawane za wystarczające:

1. freeware, czyli oprogramowanie bezpłatne, ale zamknięte;
2. source available, czyli kod możliwy do wglądu, ale bez prawa swobodnej modyfikacji i redystrybucji;
3. rozwiązania demonstracyjnie otwarte, w których publiczny jest jedynie fragment kodu;
4. rozwiązania open core, w których otwarty rdzeń wymaga zamkniętych, płatnych modułów do realnej eksploatacji w administracji;
5. usługi SaaS deklarowane jako open source, ale bez możliwości eksportu danych, konfiguracji, logów, metadanych i przejęcia utrzymania przez innego operatora.

Szczególnie ważne jest wyłączenie lub co najmniej silne ograniczenie premiowania modeli open core. W takim modelu podstawowa wersja oprogramowania bywa dostępna na licencji open source, ale funkcje krytyczne dla administracji — np. SSO, integracja z usługami katalogowymi, zaawansowany audyt, wysokodostępna architektura, klastrowanie,

retencja, archiwizacja, zarządzanie uprawnieniami czy narzędzia compliance — pozostają zamknięte i płatne. W praktyce prowadzi to do przejścia z jednego vendor lock-in do drugiego: z zależności od globalnego producenta oprogramowania własnościowego do zależności od mniejszego, ale nadal nieprzenaszalnego dostawcy.

Definicja konkursowa powinna więc obejmować zasadę, że wszystkie komponenty wymagane do produkcyjnego, bezpiecznego i zgodnego z prawem działania systemu w administracji muszą być dostępne na licencji open source albo muszą być wymienne bez utraty funkcjonalności, danych i bezpieczeństwa. Nie powinno wystarczać, że „jakiś element” rozwiązania jest otwarty. Otwartość musi dotyczyć całości wdrażanej architektury w zakresie niezbędnym do jej samodzielnego utrzymania, audytu, migracji i rozwoju.

Odpowiadając bezpośrednio na pytanie o licencję: definicja powinna odwoływać się zarówno do ogólnych cech licencji open source, jak i do konkretnych licencji lub ich grup. Nie rekomendujemy jednak zamykania konkursu wyłącznie do jednej sztywnej listy. Najlepszym rozwiązaniem jest model dwustopniowy:

- Po pierwsze, regulamin powinien zawierać definicję funkcjonalną, określającą prawa użytkownika: prawo używania w dowolnym celu, prawo analizy kodu, prawo modyfikacji, prawo kopiowania, prawo redystrybucji, prawo rozpowszechniania wersji zmodyfikowanych oraz brak dyskryminacji użytkowników, pól zastosowania i sposobów użycia.
- Po drugie, regulamin powinien zawierać załącznik licencyjny z listą licencji domyślnie akceptowalnych oraz procedurą dopuszczenia licencji równoważnych. W załączniku warto wskazać m.in. EUPL 1.2, GPL, AGPL, LGPL, MPL 2.0, Apache 2.0, MIT, BSD oraz inne licencje powszechnie uznawane w ekosystemie wolnego i otwartego oprogramowania. Identyfikacja licencji powinna odbywać się z użyciem jednoznacznych identyfikatorów SPDX, co ograniczy błędy formalne i ułatwi audyt.

Dla kodu, konfiguracji, skryptów, integracji i dokumentacji tworzonych lub finansowanych w ramach konkursu rekomendujemy szczególnie silne premiowanie licencji EUPL 1.2. Jest to licencja europejska, zaprojektowana z myślą o administracji publicznej i realiach prawnych Unii Europejskiej. Dobrze odpowiada na potrzebę ochrony rezultatów finansowanych ze środków publicznych przed późniejszym zamknięciem i odsprzedawaniem innym jednostkom administracji jako produktu własnościowego. Jednocześnie zachowuje interoperacyjność z wieloma innymi licencjami open source. W przypadku środków publicznych zasada powinna brzmieć: to, co zostało wytworzone za pieniądze publiczne i może być ponownie wykorzystane przez inne jednostki, powinno być dostępne publicznie na warunkach umożliwiających ponowne użycie.

Sama licencja oprogramowania nie wystarczy jednak do osiągnięcia suwerenności cyfrowej. Konkurs powinien uzupełnić definicję open source o wymogi dotyczące otwartych standardów, otwartych formatów danych i przenaszalności. Oprogramowanie może mieć otwarty kod, ale nadal prowadzić do praktycznego uzależnienia, jeżeli dane są przechowywane w formatach zamkniętych, API jest nieudokumentowane, dokumentacja wdrożeniowa nie istnieje, konfiguracja jest zależna od jednego integratora, a migracja do innego rozwiązania wymaga kosztownego reverse engineeringu.

Dlatego proponujemy, aby za rozwiązanie zgodne z celami konkursu uznawać tylko takie, które spełnia łącznie następujące warunki:

1. pełny kod źródłowy kluczowych komponentów jest dostępny na licencji open source;
2. licencja pozwala na używanie, analizę, modyfikację, kopiowanie i redystrybucję;
3. wdrożenie nie wymaga zamkniętych komponentów do realizacji funkcji krytycznych;
4. dokumentacja administratora, użytkownika i integratora jest dostępna w formie umożliwiającej ponowne użycie;
5. dane, konfiguracje, logi i metadane mogą być wyeksportowane w otwartych lub co najmniej dobrze udokumentowanych formatach;
6. rozwiązanie posiada otwarte, udokumentowane API tam, gdzie przewidziana jest integracja z innymi systemami;
7. możliwe jest przejęcie utrzymania przez inny podmiot bez utraty dostępu do kodu, danych i dokumentacji;
8. zamawiający otrzymuje plan wyjścia z zależności od konkretnego wykonawcy, hostingu, operatora lub producenta;
9. rozwiązanie stosuje otwarte standardy wymiany dokumentów i danych;
10. dla elementów wytworzonych w ramach konkursu zapewniono publiczną dostępność kodu, dokumentacji i materiałów wdrożeniowych.

W obszarze dokumentów biurowych szczególne znaczenie ma rozróżnienie między obsługą formatów rynkowych a formatem docelowym administracji. Z uwagi na realia urzędów konieczna jest przejściowa interoperacyjność z formatami DOCX, XLSX i PPTX. Nie powinno to jednak oznaczać utrwalenia tych formatów jako podstawy suwerenności cyfrowej. Konkurs powinien premiować stosowanie ODF jako docelowego, otwartego, edytowalnego formatu pracy i archiwizacji, przy zachowaniu importu i eksportu do formatów dominujących tam, gdzie jest to operacyjnie niezbędne. Tylko takie podejście ograniczy ryzyko przejścia z jednego uzależnienia technologicznego w inne.

Analogiczna zasada dotyczy geoinformacji. Z perspektywy środowiska GIS i społeczności QGIS szczególnie ważne jest, aby open source było łączone z otwartymi standardami danych przestrzennych. QGIS, PostgreSQL/PostGIS, GDAL/OGR, GeoServer, QGIS Server czy MapServer pokazują, że profesjonalne systemy open source mogą realnie konkurować z komercyjnymi rozwiązaniami klasy ArcGIS Enterprise czy Oracle Spatial. Ich siła wynika jednak nie tylko z otwartego kodu, ale także ze stosowania otwartych formatów i standardów, takich jak GeoPackage, WMS, WFS, OGC API czy otwarte schematy danych. To dobry przykład dla całego konkursu: administracja powinna uniezależnić się nie tylko od konkretnego producenta aplikacji, ale także od zamkniętych formatów (takich jak powszechnie stosowany w projektowaniu Autodesk DWG), zamkniętych protokołów i zamkniętych modeli utrzymania.

Podsumowując, definicja open source na potrzeby konkursu powinna być ambitna, ale praktyczna. Powinna chronić administrację przed trzema ryzykami: pozorną otwartością, nowym vendor lock-in oraz finansowaniem z pieniędzy publicznych rozwiązań, których nie da się potem ponownie wykorzystać. Najlepszym rozwiązaniem jest definicja oparta na prawach użytkownika, uzupełniona załącznikiem licencyjnym, wymogiem otwartych standardów, obowiązkiem przenaszalności danych oraz preferencją dla EUPL 1.2 przy rezultatach tworzonych za środki publiczne. Wtedy konkurs nie będzie wyłącznie testem

alternatywnych aplikacji, lecz realnym narzędziem budowy europejskiej i polskiej suwerenności cyfrowej.

2. Jakie czynniki będą służyły skuteczności wdrożeń?

Skuteczność wdrożeń open source w administracji publicznej będzie zależała przede wszystkim od tego, czy konkurs zostanie zaprojektowany jako program budowy trwałych zdolności organizacyjnych, technicznych i kompetencyjnych, a nie jako jednorazowa wymiana aplikacji. Open source nie powinien być rozumiany jako „tańszy zamiennik” narzędzi własnościowych, lecz jako element szerszej strategii suwerenności cyfrowej państwa: kontroli nad danymi, ograniczenia vendor lock-in, racjonalizacji kosztów, zwiększenia konkurencyjności rynku usług IT oraz budowy kompetencji po stronie administracji i polskich/europejskich wykonawców.

Wyniki badania MC-COI pokazują, że administracja nie startuje od zera. Znaczna część jednostek używa już jakiegoś oprogramowania open source, szczególnie w pakietach biurowych, systemach operacyjnych, poczcie i kalendarzu, grafice, zarządzaniu danymi oraz GIS. Jednocześnie dominującym modelem w kluczowych obszarach pozostaje oprogramowanie własnościowe, a największymi problemami tego modelu są koszty, zależność od dostawcy i ograniczenia licencyjne. Oznacza to, że konkurs powinien wykorzystać istniejące doświadczenia administracji, uporządkować je, ustandaryzować i przekształcić w powtarzalny model działania.

Pierwszym czynnikiem skuteczności będzie dobór właściwego zakresu pilotaży. Największy potencjał wdrożeń leży w obszarach, w których występuje duży efekt skali, powtarzalność procesów i relatywnie wysoki koszt rozwiązań własnościowych: pakiety biurowe, poczta i kalendarz, komunikacja, wideokonferencje, chmura prywatna, zarządzanie dokumentami, zarządzanie danymi oraz wybrane systemy dziedzinowe. W tych obszarach korzyści z otwartości są szczególnie wysokie, ponieważ wiele jednostek wykonuje podobne czynności, ma podobne potrzeby i mogłoby korzystać z tych samych wzorców wdrożeniowych.

Drugim czynnikiem jest etapowość. Migracja do open source nie powinna być jednorazową, gwałtowną operacją obejmującą równocześnie wszystkie elementy środowiska pracy. Lepszy jest model stopniowy: najpierw audyt stanu obecnego, następnie wybór obszarów o najlepszym stosunku korzyści do ryzyka, później pilotaż w wybranych komórkach organizacyjnych, a dopiero potem szersze wdrożenie. W wielu jednostkach rozsądną ścieżką może być rozpoczęcie od aplikacji już znanych lub częściowo obecnych w administracji, np. narzędzi biurowych, poczty, przeglądark, systemów współpracy, baz danych lub GIS. Taka metoda zmniejsza opór użytkowników i pozwala szybciej osiągnąć widoczny sukces.

Trzecim czynnikiem jest rzetelna analiza przedwdrożeniowa. Każdy pilotaż powinien zaczynać się od inwentaryzacji: używanego oprogramowania, licencji, kosztów utrzymania, formatów danych, integracji, szablonów dokumentów, makr, procedur, kont użytkowników, uprawnień, systemów dziedzinowych i wymagań bezpieczeństwa. Bez tej wiedzy wdrożenie

będzie oparte na intuicji, a nie na realnej mapie ryzyk. Taki audyt powinien również wskazać, które elementy można zastąpić szybko, które wymagają okresu współistnienia, a które powinny zostać przeniesione dopiero po zmianie procesów lub standardów danych.

Czwartym czynnikiem jest centralne wsparcie państwa. Badanie MC-COI bardzo wyraźnie pokazuje, że jednostki oczekują centralnego wsparcia technicznego, jasnych wytycznych państwowych, katalogu zweryfikowanych rozwiązań open source, finansowania migracji i szkoleń oraz księgi dobrych praktyk. Są to postulaty absolutnie kluczowe. Konkurs nie powinien pozostawiać każdej jednostki samej sobie. Równolegle do konkursu powinno działać krajowe centrum kompetencji open source dla administracji, pełniące rolę helpdesku drugiej linii, ośrodka standaryzacji, miejsca wymiany doświadczeń i źródła wzorcowych dokumentów.

Piątym czynnikiem jest katalog zweryfikowanych rozwiązań. Administracja potrzebuje nie tylko informacji, że „istnieje open source”, ale praktycznego katalogu narzędzi ocenionych pod względem dojrzałości, licencji, bezpieczeństwa, aktywności społeczności, dostępności wsparcia, integracji, dokumentacji, standardów danych i kosztów utrzymania. Katalog powinien być podzielony na klasy funkcjonalne, np. pakiet biurowy, poczta, kalendarz, wideokonferencje, komunikator, chmura plików, baza danych, GIS, helpdesk, monitoring, bezpieczeństwo, EZD, zarządzanie tożsamością. Nie powinien promować jednego wykonawcy, lecz wskazywać dojrzałe rozwiązania i warunki ich bezpiecznego użycia.

Szóstym czynnikiem jest pakietowość rozumiana jako integracja modułów, a nie jako nowy monolit. Konkurs słusznie powinien premiować rozwiązania pakietowe, ale nie powinien tworzyć zależności od jednego „superdostawcy”. Docelowy model to zintegrowane, modułowe środowisko pracy: chmura prywatna, edycja dokumentów, poczta, kalendarz, komunikacja, wideokonferencje, zarządzanie użytkownikami, SSO, backup i monitoring. Komponenty powinny być wymienne dzięki otwartym standardom i dobrze udokumentowanym API. Wtedy administracja uzyskuje realną elastyczność: może wymienić komunikator, pakiet biurowy, system plików lub bazę danych bez rozbijania całej architektury.

Siódmym czynnikiem jest oparcie wdrożeń na otwartych standardach i przenaszalności danych. Sama otwartość kodu nie wystarczy, jeśli dane pozostaną zamknięte w formatach lub API kontrolowanych przez jednego dostawcę. Skuteczne wdrożenie powinno zatem zakładać: eksport danych, dokumentacji, konfiguracji, logów i metadanych; stosowanie otwartych formatów; opisane interfejsy integracyjne; brak zależności od niejawnych komponentów; oraz plan wyjścia od konkretnego wykonawcy lub operatora. W obszarze dokumentów biurowych oznacza to stopniowe przechodzenie na ODF jako docelowy format pracy i archiwizacji, przy zachowaniu kontrolowanej interoperacyjności z formatami DOCX/XLSX/PPTX w okresie przejściowym. W obszarze danych przestrzennych oznacza to stosowanie standardów OGC, GeoPackage, WMS, WFS, OGC API i otwartych schematów danych.

Ósmym czynnikiem jest zarządzanie zmianą po stronie użytkowników. Migracja do open source jest w dużej mierze projektem organizacyjno-szkoleniowym, a dopiero później technicznym. Użytkownicy powinni dostać czas, szkolenia zadaniowe, materiały porównawcze, bazę wiedzy, szybkie wsparcie stanowiskowe i możliwość zgłaszania

problemów. Nie wystarczy jednorazowe szkolenie ogólne. Potrzebne są krótkie instrukcje typu: „jak wykonać w nowym narzędziu zadania, które dotąd wykonywano w rozwiązaniu własnościowym”. Należy też wyznaczyć lokalnych ambasadorów zmiany w urzędach, którzy będą pomagać zespołom w pierwszych tygodniach po wdrożeniu.

Dziewiątym czynnikiem jest uczciwe podejście do kompatybilności. Badanie MC-COI pokazuje, że kompatybilność z istniejącymi systemami komercyjnymi jest jedną z kluczowych barier i motywacji wyboru rozwiązań własnościowych. Nie należy tej obawy lekceważyć. Skuteczny konkurs powinien przyjąć model przejściowy: zachować interoperacyjność z dominującymi formatami i systemami tam, gdzie jest to konieczne operacyjnie, ale równocześnie konsekwentnie budować docelowe środowisko oparte na otwartych standardach. Celem nie jest chaos migracyjny, lecz stopniowe odzyskiwanie kontroli nad formatami, danymi i procesami.

Dziesiątym czynnikiem jest bezpieczeństwo rozumiane jako proces utrzymaniowy. Open source daje wielką przewagę w postaci możliwości audytu kodu, jawności zależności i niezależnej weryfikacji, ale bezpieczeństwo nie pojawia się automatycznie. Pilotáže powinny obejmować procedury aktualizacji, monitorowania podatności, zarządzania komponentami, backupu, odtwarzania, logowania działań administracyjnych, kontroli uprawnień, testów bezpieczeństwa i reagowania na incydenty. Skuteczne wdrożenie to nie tylko instalacja systemu, lecz zapewnienie zdolności do jego bezpiecznego utrzymania po zakończeniu pilotażu.

Jedenastym czynnikiem jest trwały model finansowania. Wdrożenie open source nie oznacza braku kosztów. Oznacza zmianę struktury kosztów: mniej środków na licencje i subskrypcje, więcej na analizę, migrację, szkolenia, integrację, dokumentację, bezpieczeństwo, lokalne wsparcie i rozwój kompetencji. To zmiana korzystna dla państwa i podatników, ponieważ środki mogą zasilać lokalny oraz europejski ekosystem usług, zamiast być w dużej części transferowane do globalnych dostawców technologii. Konkurs powinien zatem finansować pełny proces wdrożenia, a nie tylko techniczne uruchomienie oprogramowania.

Dwunastym czynnikiem jest obowiązek pozostawienia po pilotażu użytecznych artefaktów. Każde wdrożenie powinno zakończyć się przekazaniem do centralnego repozytorium: dokumentacji architektury, instrukcji administratora, instrukcji użytkownika, skryptów instalacyjnych, konfiguracji, opisów integracji, materiałów szkoleniowych, listy problemów, rekomendacji powdrożeniowych i planu utrzymania. Bez tego pilotaż pozostanie lokalnym eksperymentem. Z tymi artefaktami może stać się wzorem do powtórzenia w innych jednostkach.

Trzynastym czynnikiem jest wybór dojrzałych projektów i unikanie izolowanych kopii repozytorium kodu (fork'ów). Administracja powinna opierać się na projektach z aktywną społecznością, jasnym modelem utrzymania, regularnymi wydaniem, dobrą dokumentacją i dostępnością wielu wykonawców. Nie należy finansować rozwiązań, które będą działać tylko w jednej jednostce i tylko z jednym integratorem. Każde dostosowanie powinno być projektowane tak, aby mogło wrócić do głównego projektu albo zostać ponownie użyte przez inne jednostki. Warto przy okazji wprowadzić w obszarze zamówień publicznych zasadę

premiującą projekty, których wdrożenie umożliwi finansowanie komponentów dających się użyć przez inne jednostki administracji.

Czternastym czynnikiem jest wykorzystanie obszarów, w których open source już udowodnił profesjonalną dojrzałość. Szczególnie dobrym przykładem jest GIS. Ekosystem QGIS, PostgreSQL/PostGIS, GDAL/OGR, GeoServer, QGIS Server, MapServer i standardów OGC pokazuje, że wolne i otwarte oprogramowanie może być realną alternatywą dla kosztownych rozwiązań komercyjnych takich jak ArcGIS Enterprise czy Oracle Spatial. Co ważne, w badaniu MC-COI systemy GIS i geodezja są już jednym z obszarów, w których jednostki wskazują wykorzystanie open source. To pokazuje, że sukces jest możliwy tam, gdzie istnieje dojrzałe oprogramowanie, aktywna społeczność, otwarte standardy i realny rynek wsparcia. Ten model warto przenieść także na inne obszary administracji.

Piętnastym czynnikiem jest jasne poparcie kierownictwa jednostki. Migracja do open source wymaga decyzji strategicznej, a nie wyłącznie technicznej. Kierownictwo musi komunikować, że celem jest suwerenność cyfrowa, racjonalizacja wydatków, bezpieczeństwo, uniezależnienie od pojedynczych dostawców i budowa kompetencji publicznych. Bez takiego poparcia nawet dobrze przygotowane wdrożenie może zostać zablokowane przez przyzwyczajenia, obawy i lokalne interesy.

Podsumowując, skuteczność wdrożeń zapewnią: etapowość, audyt stanu wyjściowego, centralne wsparcie, katalog zweryfikowanych rozwiązań, finansowanie migracji i szkoleń, otwarte standardy, przenaszalność danych, bezpieczeństwo utrzymaniowe, zarządzanie zmianą, dokumentacja, repozytorium efektów pilotażu oraz wybór dojrzałych projektów z realnym ekosystemem wsparcia. Najważniejsze jest jednak to, aby konkurs nie produkował pojedynczych wysp sukcesu, lecz powtarzalne wzorce cyfrowo suwerennego stanowiska pracy administracji. Tylko wtedy open source stanie się trwałym elementem modernizacji państwa, a nie jednorazowym eksperymentem.

3. W jaki sposób zapewnić maksymalną replikowalność wdrażanych rozwiązań w innych jednostkach administracji publicznej?

Maksymalna replikowalność wdrażanych rozwiązań powinna być jednym z głównych celów konkursu, a nie efektem ubocznym pilotażu. Jeżeli konkurs zakończy się jedynie kilkoma lokalnymi sukcesami, których nie da się łatwo przenieść do innych jednostek, jego znaczenie systemowe będzie ograniczone. Dlatego replikowalność powinna zostać wpisana w regulamin konkursu, kryteria oceny, umowy z wykonawcami, wymagania dokumentacyjne i sposób odbioru rezultatów.

Rekomendujemy, aby każde wdrożenie finansowane w ramach konkursu było projektowane zgodnie z zasadą „reuse by default”, czyli „projektowania z myślą o ponownym użyciu”. Oznacza to, że już od pierwszego dnia pilotażu wykonawca i beneficjent powinni przygotowywać rozwiązanie nie tylko dla własnej jednostki, ale jako wzorzec możliwy do

powielenia w innych urzędach. Nie należy finansować rozwiązań „szytych na miarę” w taki sposób, że po zakończeniu projektu działają wyłącznie w jednej gminie, jednym powiecie lub jednym urzędzie i wymagają tego samego integratora do każdej kolejnej modyfikacji.

Replikowalność powinna być rozumiana w czterech warstwach: prawnej, technicznej, organizacyjnej i semantycznej:

- Warstwa prawna oznacza, że kod, konfiguracje, dokumentacja, materiały szkoleniowe, skrypty migracyjne i inne rezultaty powstałe za środki publiczne muszą być dostępne na warunkach umożliwiających ich ponowne użycie, modyfikację i rozpowszechnianie przez inne jednostki administracji. Rekomendujemy, aby nowe elementy wytworzone w ramach konkursu były co do zasady publikowane na licencji EUPL 1.2 albo innej kompatybilnej licencji open source, która chroni możliwość ponownego użycia i ogranicza ryzyko późniejszego zamknięcia rezultatów finansowanych publicznie.
- Warstwa techniczna oznacza, że wdrożenie musi dać się odtworzyć w innym środowisku w sposób możliwie zautomatyzowany. Należy premiować rozwiązania oparte na konteneryzacji, Infrastructure as Code, wersjonowanych plikach konfiguracyjnych, opisanych zależnościach, automatycznych testach instalacji i udokumentowanych procedurach aktualizacji. Wdrożenie nie powinno polegać na ręcznej, niepowtarzalnej konfiguracji wykonanej przez jednego administratora lub jednego integratora. Jeżeli inna jednostka ma powtórzyć wdrożenie, powinna otrzymać kompletną instrukcję, repozytorium konfiguracji i jasny opis wymagań infrastrukturalnych.
- Warstwa organizacyjna oznacza, że wraz z technologią musi zostać przeniesiony model działania. Inna jednostka powinna wiedzieć, jakie role są potrzebne, kto odpowiada za użytkowników, kto za aktualizacje, kto za backup, kto za bezpieczeństwo, kto za zgłoszenia, a kto za kontakt z wykonawcą lub centrum wsparcia. Replikowalność nie dotyczy więc wyłącznie kodu, ale również procedur, matryc odpowiedzialności, wzorcowych regulaminów, instrukcji dla administratorów, scenariuszy migracji i materiałów szkoleniowych.
- Warstwa semantyczna oznacza, że dane i dokumenty muszą być wymieniane w otwartych, dobrze opisanych formatach i zgodnie z nazwanymi standardami. Oprogramowanie może mieć otwarty kod, ale jeżeli dane pozostaną zamknięte w niestandardowym formacie, replikowalność będzie pozorna. Dlatego konkurs powinien premiować ODF jako docelowy format dokumentów edytowalnych, otwarte API, standardy integracyjne, jawne schematy danych, eksport konfiguracji i możliwość migracji do innego rozwiązania. W geoinformacji analogiczną rolę powinny pełnić standardy OGC, GeoPackage, WMS, WFS, OGC API, otwarte schematy danych oraz narzędzia takie jak QGIS i PostgreSQL/PostGIS.

Kluczowym narzędziem zapewnienia replikowalności powinno być centralne, publiczne repozytorium rezultatów konkursu. Każde wdrożenie powinno po zakończeniu pozostawić w takim repozytorium co najmniej:

1. kod źródłowy elementów wytworzonych w ramach projektu;
2. skrypty instalacyjne i migracyjne;
3. pliki konfiguracyjne oraz opis parametrów środowiskowych;

4. dokumentację architektury;
5. instrukcję instalacji i odtworzenia środowiska;
6. instrukcję administratora;
7. instrukcję użytkownika;
8. opis integracji z systemami dziedzinowymi;
9. opis użytych standardów i formatów danych;
10. wzorcowe materiały szkoleniowe;
11. listę napotkanych problemów i sposobów ich rozwiązania;
12. procedurę aktualizacji;
13. procedurę backupu i odtworzenia;
14. plan wyjścia od konkretnego dostawcy, operatora lub integratora;
15. rekomendacje dla kolejnych jednostek chcących powtórzyć wdrożenie.

Repozytorium powinno być prowadzone w sposób uporządkowany, z wersjonowaniem, publicznym śledzeniem wątków (issue trackerem), opisem statusu projektów, oznaczeniem dojrzałości i informacją, które wdrożenia zostały już powtórzone w innych jednostkach. Nie powinno być biernym archiwum plików, lecz żywą platformą współpracy administracji publicznej. Warto rozważyć model inspirowany hiszpańskim Centro de Transferencia de Tecnología, gdzie administracja ma mechanizm wyszukiwania i ponownego użycia rozwiązań już wytworzonych w sektorze publicznym.

Warto wspomnieć również o modelu fińskim. W Finlandii wdrożono pionierski model konsolidacji pod nazwą COSS National QGIS Collaboration. COSS to organizacja non-profit promująca otwarte oprogramowanie, otwarte dane, otwarte standardy i interfejsy API. Na arenie międzynarodowej COSS jest znane jako jedno z najstarszych i najbardziej aktywnych centrów otwartości. Działalność COSS opiera się na dalekosiężnej współpracy, komunikacji i tworzeniu sieci. Celem COSS jest wzmocnienie konkurencyjności fińskich sektorów przemysłu opartego na oprogramowaniu, wspieranie wzrostu i internacjonalizacji przedsiębiorstw zajmujących się oprogramowaniem typu open source oraz wzmocnienie rozwoju fińskiego społeczeństwa informacyjnego przy wsparciu otwartych technologii i społeczności. Fińskie Centrum Otwartych Systemów i Rozwiązań (COSS) zrzeszyło kluczowe instytucje państwowe, w tym Narodowy Instytut Geodezji (National Land Survey of Finland), Centrum Leśne, Agencję Infrastruktury Transportowej oraz Instytut Środowiska (Syke), w jeden silny podmiot reprezentujący interesy Finlandii w zakresie otwartego oprogramowania. Dzięki temu poszczególne agencje nie muszą budować odrębnych kanałów komunikacji ani umów finansowych; COSS centralnie zarządza składkami i wpływa na kierunki rozwoju oprogramowania, stymulując jednocześnie wdrażanie otwartych rozwiązań w strukturach administracji. Rozwiązanie to gwarantuje pełne bezpieczeństwo dostaw oraz odporność technologiczną państwa. Konkurs powinien wymagać, aby przed rozpoczęciem budowy lub zakupu niestandardowego komponentu beneficjent sprawdził, czy podobne rozwiązanie nie istnieje już w katalogu publicznym. Jeżeli istnieje, powinien uzasadnić, dlaczego nie może go użyć lub dostosować. Taka zasada ograniczy powielanie kosztów i tworzenie wielu lokalnych rozwiązań realizujących te same funkcje. Administracja nie powinna wielokrotnie płacić za rozwiązanie tego samego problemu.

Bardzo ważne jest ograniczenie zjawiska „administracyjnego płątka śniegu”. Każda jednostka ma pewne lokalne uwarunkowania, ale wiele procesów administracyjnych jest podobnych. Nadmierne dopasowywanie systemu do historycznych, lokalnych przyzwyczajień

proceeds to solutions, which cannot be transferred. The competition should reward standardization and parametrization instead of hard-coded local exceptions. Differences between units should be managed through configuration, templates, roles, dictionaries, profiles and modules, and not through creating separate forks for each office.

Reproducibility also requires referential architectures. The Ministry or an open source center should prepare several model architectures, e.g. for a small municipality, a county, a city or a central unit. These models should take into account the number of users, different local IT competences, different levels of accessibility and different infrastructure variants: on-premise, private cloud, shared infrastructure of several units or hybrid model. Thanks to this, beneficiaries will not have to start from scratch.

It is important that the competition distinguishes between packages and monoliths. A package solution is preferred, because the user needs a unified work environment: files, documents, mail, calendar, communicator, video conference, identity, permissions and backup. It should not be a single locked system. The best model is an integrated set of modules connected by open standards and managed together. If in a few years a better component for mail, communicator, video conference or document editing appears, the administration should be able to replace it without rebuilding the whole environment.

Every project should have a migration strategy. Reproducibility and the possibility of migration are linked. If a unit cannot leave a specific provider, it should be able to move to another unit. The migration plan should describe how to export data, documents, accounts, permissions, configurations, logs and metadata; how to recreate the system at another operator; how to change the integrator; how to replace a component; and what are the minimal conditions for maintaining the solution after the pilot project ends.

The competition should also require measurable indicators of reproducibility. It is not enough to declare that the solution "can be reproduced". One should evaluate, for example:

1. there is public documentation of the implementation;
2. the environment can be recreated from a repository;
3. the configuration is versioned;
4. data can be exported;
5. the implementation does not require locked critical components;
6. another contractor can take over the maintenance;
7. the instructions are understandable for the administrator of another unit;
8. training materials have been prepared;
9. the costs of maintenance are described;
10. a test recreation of the environment has been performed;
11. minimal and recommended infrastructure requirements are indicated;
12. dependencies on external services are described;
13. the solution has a plan for updates and security.

A good mechanism would be to require a "reproducibility package" as the final product of each pilot project. Such a package should be evaluated just as seriously as

samo uruchomienie systemu. W jego skład powinny wejść: dokumentacja, kod, konfiguracje, procedury, materiały szkoleniowe, wzorcowe zapisy do zamówień publicznych, analiza kosztów, analiza ryzyk, plan migracji i plan utrzymania. Inna jednostka powinna móc rozpocząć własne wdrożenie od tego pakietu, zamiast od pustej kartki.

Warto również utworzyć katalog dobrych praktyk i katalog antywzorców. Administracja powinna wiedzieć nie tylko, co działa, ale także czego unikać: rozwiązań zależnych od jednego integratora, braku dokumentacji, lokalnych forków bez planu utrzymania, zamkniętych modułów enterprise, niestandardowych formatów danych, ręcznej konfiguracji bez automatyzacji, braku testów odtworzeniowych i braku planu migracji. Uczenie się na problemach pierwszych pilotaży może znacząco obniżyć koszty kolejnych wdrożeń.

Replikowalność powinna też obejmować zamówienia publiczne. Wiele jednostek może nie mieć kompetencji do samodzielnego opisanie przedmiotu zamówienia na open source. Dlatego konkurs powinien dostarczyć wzorcowe klauzule do SWZ i umów: wymagania dotyczące licencji, przekazania kodu, dokumentacji, otwartych formatów, API, przenaszalności danych, bezpieczeństwa, dostępności repozytorium, planu wyjścia i braku zależności od zamkniętych komponentów krytycznych. Bez takich wzorców każda jednostka będzie powtarzała te same błędy prawne i techniczne.

Istotnym elementem jest także wspólnota praktyków. Replikowalność nie powstaje tylko przez dokumenty, lecz przez ludzi. Potrzebne są regularne spotkania beneficjentów, administratorów, wykonawców, ekspertów bezpieczeństwa i społeczności open source. Jednostki, które zakończyły pilotaż, powinny wspierać jednostki rozpoczynające wdrożenie. Można rozważyć model mentorstwa między urzędami: jednostka A, która wdrożyła dany pakiet, pomaga jednostce B w powtórzeniu wdrożenia. Taki mechanizm buduje kompetencje w administracji i zmniejsza zależność od zewnętrznych dostawców.

Z perspektywy GIS istnieje już dobry wzorzec replikowalności. Ekosystem QGIS, PostgreSQL/PostGIS, GDAL/OGR, GeoServer, MapServer i standardów OGC pokazuje, że otwarte oprogramowanie może tworzyć profesjonalny, powtarzalny stos technologiczny wykorzystywany w wielu organizacjach. QGIS nie jest zamkniętym produktem jednego dostawcy, lecz narzędziem rozwijanym przez społeczność, firmy, administracje i instytucje naukowe. PostgreSQL/PostGIS pozwala budować otwarte bazy danych przestrzennych bez zależności od rozwiązań klasy Oracle Spatial. Standardy OGC umożliwiają wymianę usług i danych między różnymi systemami. To model, który warto przenieść na inne obszary: otwarte narzędzie, otwarta baza, otwarte formaty, wielu dostawców wsparcia, aktywna społeczność i możliwość samodzielnego rozwoju.

Replikowalność wymaga także myślenia o skali krajowej. Jeżeli pilotaże pokażą dobre efekty, państwo powinno móc uruchomić kolejną falę wdrożeń bez ponownego projektowania konkursu. Dlatego już pierwsza edycja powinna zostać zaplanowana jako fundament krajowego programu: z repozytorium, katalogiem, centrum kompetencji, metodyką, wzorcami zamówień, pakietami replikacyjnymi i systemem ewaluacji. W przeciwnym razie wiedza z pilotaży rozproszy się po pojedynczych jednostkach i wykonawcach.

Podsumowując, maksymalną replikowalność zapewnią: obowiązek projektowania z myślą o ponownym użyciu, centralne repozytorium rezultatów, otwarte licencje, otwarte standardy, Infrastructure as Code, konteneryzacja, dokumentacja, materiały szkoleniowe, wzorcowe zamówienia publiczne, strategia wyjścia, katalog zweryfikowanych rozwiązań, publiczny system śledzenia zagadnień (issue tracker), społeczność praktyków i mierzalny pakiet replikacyjny jako produkt końcowy każdego pilotażu. Celem powinno być to, aby każde wdrożenie finansowane ze środków publicznych zmniejszało koszt kolejnego wdrożenia w innej jednostce. Dopiero wtedy konkurs stanie się narzędziem systemowej modernizacji administracji, a nie zbiorem odrębnych lokalnych eksperymentów.

4. Jakie czynniki powinny być brane pod uwagę przez podmioty realizujące pilotaże w zakresie bezpieczeństwa wdrażanych rozwiązań?

W obszarze bezpieczeństwa podstawowym założeniem powinno być odejście od uproszczonego sporu, czy open source jest „z definicji” bezpieczniejszy albo mniej bezpieczny od oprogramowania własnościowego. Oprogramowanie open source jest bezpieczne wtedy, gdy jest prawidłowo wybrane, wdrożone, skonfigurowane, aktualizowane, monitorowane i utrzymywane. Jego przewagą jest jawność kodu, możliwość niezależnego audytu, brak konieczności ślepego zaufania jednemu producentowi oraz możliwość przejęcia utrzymania przez inny podmiot. Te zalety trzeba jednak połączyć z profesjonalnym zarządzaniem bezpieczeństwem, ponieważ sama otwartość kodu nie zastąpi procedur, kompetencji i odpowiedzialności.

Wyniki badania MC-COI pokazują, że bezpieczeństwo jest jednym z obszarów, w których administracja zachowuje ostrożność wobec open source. W obszarze „bezpieczeństwo — nie antywirus” dominuje model własnościowy, a ocena bezpieczeństwa OSS w porównaniu z oprogramowaniem własnościowym jest niejednoznaczna. Duża część respondentów nie potrafi jednoznacznie ocenić, które podejście jest bezpieczniejsze. To bardzo ważny sygnał dla konstrukcji konkursu: bezpieczeństwo nie może być opisane hasłowo. Musi zostać przełożone na obiektywne, mierzalne i audytowalne wymagania.

Pierwszym czynnikiem, który powinien być brany pod uwagę, jest zgodność z obowiązującymi wymaganiami prawa i polityk bezpieczeństwa administracji. Pilotaże powinny uwzględniać w szczególności KRI, RODO, ustawę o krajowym systemie cyberbezpieczeństwa, wymogi wynikające z NIS2 oraz nadchodzące obowiązki związane z europejskim Cyber Resilience Act. Nie chodzi jednak o deklarację zgodności, lecz o konkretne artefakty: analizę ryzyka, inwentaryzację zasobów, procedury aktualizacji, procedury nadawania uprawnień, politykę haseł i MFA, model backupu, plan odtworzenia, rejestr incydentów, logowanie działań administracyjnych i okresowe audyty bezpieczeństwa.

Drugim czynnikiem jest inwentaryzacja stanu wyjściowego. Przed wdrożeniem należy znać aktualne środowisko: używane oprogramowanie, wersje systemów, licencje, zależności, integracje, konta, role, uprawnienia, lokalizacje danych, przepływy danych, formaty plików, usługi zewnętrzne, systemy dziedziczne i istniejące zabezpieczenia. Bez

takiej mapy nie da się wiarygodnie ocenić ryzyka migracji ani zaprojektować bezpiecznego wdrożenia. Jest to szczególnie ważne, ponieważ część jednostek posiada tylko częściowo aktualną ewidencję licencji i oprogramowania, co może utrudniać ocenę podatności i zależności.

Trzecim czynnikiem jest dojrzałość wybieranego projektu open source. Podmiot realizujący pilotaż powinien ocenić, czy projekt ma aktywną społeczność, regularne wydania, publiczny system zgłaszania błędów, przejrzysty proces reagowania na podatności, dokumentację bezpieczeństwa, historię aktualizacji, wielu maintainerów oraz dostępność profesjonalnego wsparcia. Należy unikać sytuacji, w której administracja wdraża atrakcyjny, ale porzucony projekt z małą liczbą opiekunów, bez aktualizacji i bez realnego ekosystemu utrzymania. W przypadku projektów krytycznych trzeba oceniać także tzw. bus factor, czyli ryzyko zależności od bardzo małej liczby osób utrzymujących projekt.

Czwartym czynnikiem jest bezpieczeństwo łańcucha dostaw oprogramowania. Współczesne systemy open source składają się często z wielu zależności: bibliotek, obrazów kontenerów, pakietów systemowych, modułów językowych i komponentów pobieranych z repozytoriów. Każdy z tych elementów może wprowadzać podatność. Dlatego pilotaże powinny wymagać stosowania Software Bill of Materials, czyli SBOM, najlepiej w standardowych formatach takich jak SPDX lub CycloneDX. SBOM powinien być generowany automatycznie i aktualizowany przy każdej istotnej zmianie systemu. Dzięki temu w razie ujawnienia podatności wiadomo, które jednostki i które komponenty są narażone.

Piątym czynnikiem jest Software Composition Analysis, czyli automatyczna analiza komponentów i zależności. Wdrożenia powinny korzystać z narzędzi wykrywających znane podatności CVE, przestarzałe biblioteki, ryzykowne licencje, porzucone zależności, niepodpisane pakiety i obrazy pochodzące z niezweryfikowanych źródeł. Nie wystarczy, że system „działa”. Musi być wiadomo, z czego jest zbudowany, skąd pochodzą komponenty, kto je utrzymuje i jakie ryzyka są z nimi związane.

Szóstym czynnikiem jest bezpieczne pozyskiwanie komponentów. Należy preferować oficjalne repozytoria, podpisane pakiety, weryfikację integralności kryptograficznej, obrazy kontenerów z zaufanych źródeł oraz wewnętrzne, zatwierdzone repozytoria administracji lub operatora. Trzeba ograniczać ryzyka takie jak typosquatting, dependency confusion, przejęcie pakietu, użycie porzuconych bibliotek, pobieranie komponentów z prywatnych kont bez kontroli oraz brak weryfikacji sum kontrolnych. W środowisku administracji publicznej nie powinno być akceptowalne wdrożenie oparte na przypadkowym pobieraniu pakietów z internetu bez kontroli źródła i wersji.

Siódmym czynnikiem jest zarządzanie podatnościami. Każdy pilotaż powinien mieć procedurę monitorowania podatności, oceniania ich wpływu, priorytetyzacji, testowania poprawek i wdrażania aktualizacji. Wykonawca powinien wskazać osobę lub zespół odpowiedzialny za triage podatności, kanał zgłaszania błędów bezpieczeństwa, SLA dla podatności krytycznych, wysokich i średnich oraz zasady komunikacji z jednostką. Dobrą praktyką jest uczestnictwo w coordinated vulnerability disclosure, czyli skoordynowanym ujawnianiu podatności, oraz utrzymywanie jasnego procesu zgłoszeń dla administratorów i użytkowników.

Ósmym czynnikiem jest polityka aktualizacji. Open source daje możliwość szybkiego reagowania, ale tylko wtedy, gdy istnieje proces aktualizacyjny. Pilotaż powinien określać, jak często wykonywane są aktualizacje, kto je zatwierdza, jak są testowane, jak wygląda środowisko testowe, jak przebiega rollback, jak dokumentuje się zmiany i jak komunikuje się przerwy techniczne. Szczególną uwagę należy poświęcić systemom wystawionym do internetu, usługom chmurowym, komunikatorom, wideokonferencjom, poczcie, systemom tożsamości oraz komponentom odpowiedzialnym za uwierzytelnianie i szyfrowanie.

Dziewiątym czynnikiem jest bezpieczna konfiguracja. Wiele incydentów nie wynika z błędów w kodzie, lecz z błędnej konfiguracji. Dlatego wdrożenie powinno obejmować hardening systemów, zamykanie nieużywanych usług, zasadę najmniejszych uprawnień, poprawne ustawienie TLS, wymuszenie MFA tam, gdzie jest to uzasadnione, bezpieczne zarządzanie sekretami, segmentację sieci, kontrolę dostępu administracyjnego, ograniczenie kont uprzywilejowanych i regularny przegląd uprawnień. Konfiguracja powinna być wersjonowana i możliwa do audytu, najlepiej w modelu Infrastructure as Code.

Dziesiątym czynnikiem jest zarządzanie tożsamością i dostępem. Bezpieczne wdrożenie open source w administracji wymaga spójnego modelu IAM: centralnego zarządzania użytkownikami, rolami, grupami, uprawnieniami, cyklem życia kont, SSO, MFA, integracji z istniejącymi usługami katalogowymi oraz procedurą szybkiego odbierania uprawnień pracownikom odchodzącym lub zmieniającym stanowisko. Tożsamość i dostęp są szczególnie krytyczne w środowiskach pakietowych, gdzie użytkownik korzysta z poczty, dokumentów, komunikatora, wideokonferencji i chmury plików.

Jedenastym czynnikiem jest backup i odtwarzanie. Każdy pilotaż powinien mieć jasny model kopii zapasowych, retencji, szyfrowania backupów, separacji kopii od środowiska produkcyjnego oraz testów odtwarzania. Backup, którego nigdy nie odtworzono testowo, nie powinien być traktowany jako wystarczające zabezpieczenie. Należy określić RPO i RTO, czyli akceptowalną utratę danych i czas odtworzenia usługi. W administracji szczególnie ważne jest także zabezpieczenie przed ransomware, błędami administratora oraz usunięciem danych przez konto z nadmiernymi uprawnieniami.

Dwunastym czynnikiem jest logowanie, monitoring i audyt. Systemy wdrażane w pilotażu powinny umożliwiać logowanie działań administracyjnych, prób logowania, zmian uprawnień, dostępu do danych wrażliwych, błędów systemowych i zdarzeń bezpieczeństwa. Logi powinny być chronione przed modyfikacją, retencjonowane zgodnie z polityką bezpieczeństwa i możliwe do analizy przez administratorów lub SOC. Równocześnie logowanie musi być zgodne z zasadą minimalizacji danych i RODO — nie należy zbierać więcej danych osobowych, niż jest to potrzebne do bezpieczeństwa i rozliczalności.

Trzynastym czynnikiem jest ochrona danych osobowych i danych urzędowych. Wdrożenia powinny uwzględniać klasyfikację danych, minimalizację, szyfrowanie transmisji, kontrolę dostępu, separację środowisk, retencję, usuwanie danych, anonimizację lub pseudonimizację tam, gdzie jest to uzasadnione, oraz ocenę skutków dla ochrony danych, jeżeli charakter przetwarzania tego wymaga. Dotyczy to szczególnie poczty, chmury plików, komunikatorów, wideokonferencji, EKD, systemów GIS i baz danych.

Czternastym czynnikiem jest bezpieczeństwo integracji. W administracji systemy rzadko działają w izolacji. Wdrożenia open source będą integrowane z EKD, usługami

katalogowymi, pocztą, systemami finansowo-kadrowymi, rejestrami publicznymi, platformami usługowymi i systemami dziedzinowymi. Każda integracja powinna mieć opisane API, mechanizmy uwierzytelniania, zakres przekazywanych danych, logowanie, limity, sposób rotacji kluczy, procedurę wyłączenia i plan awaryjny. Integracje nie powinny opierać się na nieudokumentowanych obejściach.

Piętnastym czynnikiem jest odporność na pozorną suwerenność i open core. System może być przedstawiany jako open source, ale funkcje krytyczne dla bezpieczeństwa — audyt, SSO, MFA, HA, szyfrowanie, retencja, archiwizacja, zarządzanie uprawnieniami — mogą być dostępne tylko w zamkniętych modułach. Taki model osłabia bezpieczeństwo i zwiększa lock-in. Konkurs powinien wymagać, aby komponenty krytyczne dla bezpieczeństwa były otwarte albo co najmniej wymienne bez utraty danych i funkcjonalności. Nie należy budować bezpieczeństwa administracji na elementach, których nie można audytować, przenieść lub zastąpić.

Szesnastym czynnikiem jest zapewnienie kompetencji i wsparcia. Obawy o bezpieczeństwo OSS często wynikają nie z samego modelu otwartego kodu, lecz z braku pewności, kto będzie odpowiadał za utrzymanie, aktualizacje i incydenty. Dlatego każdy pilotaż powinien mieć wskazany model wsparcia technicznego: pierwsza linia w jednostce, druga linia u wykonawcy lub centrum kompetencji, eskalacja incydentów, czasy reakcji, procedury awaryjne i dokumentacja dla administratorów. Administracja powinna kupować nie tylko wdrożenie, ale zdolność bezpiecznego utrzymania.

Siedemnastym czynnikiem jest testowanie bezpieczeństwa. Przed produkcyjnym uruchomieniem należy przeprowadzić przegląd konfiguracji, skan podatności, testy aktualizacji, testy odtworzenia, przegląd uprawnień i analizę ryzyka. W przypadku systemów wystawionych do internetu lub przetwarzających dane wrażliwe należy rozważyć testy penetracyjne. Testy powinny być powtarzane po istotnych zmianach architektury, aktualizacjach głównych wersji i dodaniu nowych integracji.

Osiemnastym czynnikiem jest transparentność. Jedną z przewag open source jest możliwość jawnego dokumentowania ryzyka, zależności, podatności i poprawek. Pilotaż powinien tworzyć dokumentację, która może pomóc kolejnym jednostkom: listę rekomendowanych konfiguracji, listę typowych błędów, procedurę hardeningu, wzorcowy model uprawnień, checklistę bezpieczeństwa, opis incydentów testowych i rekomendacje powdrożeniowe. Dzięki temu każdy pilotaż zwiększa bezpieczeństwo kolejnych wdrożeń.

Dziewiętnastym czynnikiem jest specyfika danych przestrzennych i GIS. Systemy GIS w administracji mogą przetwarzać dane o infrastrukturze krytycznej, sieciach technicznych, planowaniu przestrzennym, środowisku, transporcie, zarządzaniu kryzysowym, mieniu publicznym i lokalizacji obiektów wrażliwych. Wdrażając rozwiązania takie jak QGIS, PostgreSQL/PostGIS, GeoServer czy usługi OGC, należy rozróżniać dane publiczne, wewnętrzne, chronione i wrażliwe. Otwarte standardy nie oznaczają automatycznie publicznego dostępu do wszystkiego. Należy precyzyjnie zarządzać uprawnieniami do warstw, usług WMS/WFS/OGC API, eksportów, metadanych i publikacji mapowej. QGIS i PostgreSQL/PostGIS są bardzo dobrym przykładem dojrzałego open source, ale tak jak każde profesjonalne narzędzie wymagają właściwego modelu dostępu, backupu, aktualizacji i audytu.

Dwudziestym czynnikiem jest plan wyjścia i ciągłość działania. Bezpieczeństwo obejmuje także możliwość zmiany wykonawcy, operatora lub komponentu bez utraty kontroli nad systemem. Dlatego wdrożenie powinno zawierać plan eksportu danych, konfiguracji, kont, uprawnień, logów i dokumentacji. W przypadku incydentu, upadłości wykonawcy lub utraty wsparcia administracja powinna móc odtworzyć usługę u innego operatora. Brak planu wyjścia jest ryzykiem bezpieczeństwa, a nie tylko ryzykiem organizacyjnym.

Podsumowując, podmioty realizujące pilotaże powinny traktować bezpieczeństwo jako proces obejmujący cały cykl życia rozwiązania: wybór projektu, analizę ryzyka, wdrożenie, konfigurację, aktualizacje, monitorowanie, reagowanie na podatności, backup, audyt i możliwość migracji. Minimalny pakiet bezpieczeństwa powinien obejmować inwentaryzację, analizę ryzyka, SBOM, SCA, politykę aktualizacji, procedurę zarządzania podatnościami, bezpieczne repozytoria komponentów, hardening, IAM, MFA, logowanie, backup, testy odtworzenia, testy bezpieczeństwa, dokumentację i jasny model odpowiedzialności. Dopiero taki zestaw wymagań pozwoli wykorzystać zalety open source — jawność, audytowalność, niezależność i przenaszalność — bez popadania w fałszywe poczucie bezpieczeństwa.

5. W jaki sposób najskuteczniej ułatwić pracownikom podmiotów realizujących pilotaże przejście z rozwiązań własnościowych na open source?

Najskuteczniejsze ułatwienie pracownikom przejścia z rozwiązań własnościowych na open source wymaga potraktowania migracji jako procesu zarządzania zmianą, a nie jako zadania polegającego na instalacji nowych aplikacji. W praktyce największym wyzwaniem nie jest zwykle sam brak funkcji w oprogramowaniu, ale przyzwyczajenia użytkowników, obawa przed utratą sprawności pracy, niepewność co do kompatybilności dokumentów oraz brak szybkiego wsparcia w pierwszych tygodniach po zmianie. Dlatego konkurs powinien finansować i wymagać pełnego programu adopcji użytkowników, obejmującego komunikację, szkolenia, wsparcie stanowiskowe, materiały instruktażowe, lokalnych ambasadorów zmiany i okres przejściowy.

Pierwszym warunkiem skutecznej migracji jest jasne wyjaśnienie celu. Pracownicy administracji powinni rozumieć, że przejście na open source nie jest eksperymentem ani prostym poszukiwaniem „darmowych zamienników”, lecz elementem budowy cyfrowej suwerenności państwa i Europy. Chodzi o ograniczenie zależności od pojedynczych dostawców, zmniejszenie kosztów licencji i subskrypcji, zwiększenie kontroli nad danymi publicznymi, rozwój kompetencji administracji oraz możliwość kierowania większej części środków publicznych na lokalne wsparcie, szkolenia, bezpieczeństwo i rozwój usług, zamiast na stałe opłaty licencyjne dla globalnych dostawców. Taka komunikacja jest ważna, bo użytkownik, który rozumie sens zmiany, znacznie łatwiej akceptuje przejściowe niedogodności.

Drugim warunkiem jest audyt rzeczywistych sposobów pracy przed wdrożeniem. Przed migracją należy zbadać, jak pracownicy faktycznie używają obecnych narzędzi: jakie

tworzą dokumenty, jakich szablonów używają, gdzie występują makra, jakie procesy zależą od arkuszy kalkulacyjnych, jakie są obiegi dokumentów, jakie formaty przychodzą z zewnątrz, jakie są integracje z EZD, pocztą, kalendarzem, systemami finansowymi, GIS lub rejestrami publicznymi. Bez takiej wiedzy szkolenia będą zbyt ogólne, a migracja może uderzyć w nieoczywiste, ale krytyczne elementy codziennej pracy.

Trzecim warunkiem jest etapowość. Nie należy wymuszać jednoczesnego przejścia na wszystkie nowe narzędzia. Skuteczniejsze jest podejście stopniowe: najpierw narzędzia najmniej ryzykowne i najłatwiejsze do przyjęcia, potem obszary bardziej złożone. Przykładowo jednostka może rozpocząć od przeglądarki, wybranych narzędzi biurowych, poczty, kalendarza, QGIS, narzędzi do grafiki, baz danych lub systemów współpracy, zanim przejdzie do głębszej przebudowy całego stanowiska pracy. Etapowość pozwala budować zaufanie i kompetencje, zamiast tworzyć wrażenie nagłej rewolucji.

Czwartym warunkiem jest uczciwe podejście do kompatybilności. Badania i praktyka pokazują, że jedną z największych barier dla open source są wymagania kompatybilności z dotychczasowymi systemami komercyjnymi. Nie należy tej bariery bagatelizować. W okresie przejściowym pracownicy muszą mieć możliwość obsługi dokumentów i danych otrzymywanych od innych instytucji, obywateli i przedsiębiorców w popularnych formatach. Jednocześnie należy konsekwentnie budować docelowy model oparty na otwartych standardach, np. ODF dla dokumentów edytowalnych oraz standardach OGC i otwartych formatach danych w geoinformacji. Należy więc odróżnić interoperacyjność przejściową od utrwalania zamkniętych formatów jako standardu państwa.

Piątym warunkiem jest przygotowanie szablonów i dokumentów przed migracją. W wielu urzędach największe problemy nie wynikają z samego edytora tekstu, lecz z historycznych szablonów, niestandardowego formatowania, starych makr, formularzy, arkuszy i dokumentów wielokrotnie kopiowanych przez lata. Przed wdrożeniem należy przejrzeć najważniejsze wzory pism, formularze, arkusze kalkulacyjne, prezentacje i szablony. Część z nich trzeba przekonwertować, część uprościć, część zastąpić nowymi wzorcami opartymi na otwartym formacie. Konkurs powinien finansować taki przegląd, ponieważ bez niego użytkownicy będą utożsamiali problemy starych dokumentów z wadami nowego oprogramowania.

Szóstym warunkiem są szkolenia zadaniowe. Tradycyjne szkolenie typu „omówienie funkcji programu” jest niewystarczające. Pracownicy powinni uczyć się na przykładach odpowiadających ich codziennym obowiązkom: przygotowanie pisma, praca z tabelą, korespondencja seryjna, podpisywanie i eksport dokumentu, obsługa kalendarza, udostępnianie plików, prowadzenie wideokonferencji, praca na wspólnym dokumencie, przygotowanie mapy, eksport danych, publikacja warstwy GIS. Najlepszy format to krótkie moduły „jak wykonać moje zadanie w nowym narzędziu” poprzedzone gruntownym audytem potrzeb, a nie wielogodzinne, abstrakcyjne prezentacje.

Siódmy warunek to materiały porównawcze. Użytkownicy potrzebują prostych instrukcji pokazujących, jak w nowym narzędziu wykonać czynności znane z dotychczasowych rozwiązań własnościowych. Powinny powstać krótkie karty pracy, filmy, checklisty i FAQ: „jak zrobić w LibreOffice to, co robiłem w Microsoft Office”, „jak zarządzać kalendarzem w nowym systemie”, „jak udostępnić plik w chmurze prywatnej”, „jak prowadzić

wideokonferencję”, „jak przygotować mapę w QGIS zamiast w komercyjnym GIS”. Tego typu materiały znacząco zmniejszają lęk przed zmianą.

Ósmy warunek to lokalni ambasadorzy zmiany. W każdej jednostce i w większych wydziałach warto wskazać osoby, które wcześniej przejdą rozszerzone szkolenie, przetestują narzędzia, pomogą w dostosowaniu materiałów do lokalnych procesów i będą pierwszym kontaktem dla współpracowników. Ambasadorzy zmiany są szczególnie skuteczni, ponieważ pracownicy często chętniej proszą o pomoc osobę z własnego zespołu niż zewnętrznego konsultanta. Konkurs powinien premiować projekty, które budują takie lokalne kompetencje, zamiast opierać się wyłącznie na wsparciu wykonawcy.

Dziewiątym warunkiem jest intensywne wsparcie w pierwszym okresie po wdrożeniu. Najwięcej frustracji pojawia się w pierwszych dniach i tygodniach, gdy użytkownik musi wykonać pilne zadanie w nowym środowisku. W tym okresie potrzebne są dyżury ekspertów, szybka ścieżka zgłoszeń, czat lub infolinia wewnętrzna, możliwość krótkiej konsultacji ekranowej oraz obecność wsparcia na miejscu w większych jednostkach. Brak szybkiej pomocy prowadzi do powrotu do starych narzędzi, obchodzenia procedur i negatywnych opinii o całym wdrożeniu.

Dziesiątym warunkiem jest centralne wsparcie techniczne i eksperckie. Jednostki samorządowe, zwłaszcza mniejsze gminy, nie powinny być pozostawione same sobie. Potrzebne jest rządowe centrum kompetencji open source dla administracji, które zapewni drugą linię wsparcia, wzorcową dokumentację, katalog zweryfikowanych rozwiązań, materiały szkoleniowe, odpowiedzi na pytania licencyjne, wytyczne dotyczące bezpieczeństwa i przykłady udanych wdrożeń. Takie centrum powinno wspierać zarówno pracowników merytorycznych, jak i lokalne zespoły IT.

Jedenastym warunkiem jest wsparcie dla administratorów. Migracja użytkowników nie powiedzie się, jeśli lokalne IT nie będzie rozumiało nowych narzędzi. Administratorzy potrzebują osobnych, technicznych szkoleń z instalacji, konfiguracji, aktualizacji, backupu, monitoringu, zarządzania kontami, uprawnieniami, integracją z katalogiem, bezpieczeństwem i procedurami awaryjnymi. Należy też przygotować dokumentację administratora oraz możliwość konsultacji z ekspertami. W wielu jednostkach to lokalny informatyk będzie osobą, od której zależy społeczne powodzenie migracji.

Dwunastym warunkiem jest ograniczenie podwójnej pracy. W okresie przejściowym czasem trzeba utrzymywać dwa środowiska, ale nie może to trwać zbyt długo ani być chaotyczne. Należy jasno określić, które procesy są już realizowane w nowym narzędziu, które pozostają tymczasowo w starym, a kiedy nastąpi wygaszenie starego modelu. Bez takiej decyzji pracownicy będą tworzyć dokumenty w dwóch standardach, dublować dane i powielać błędy. Migracja powinna mieć harmonogram, właścicieli procesów i jasne kamienie milowe.

Trzynastym warunkiem jest dobór dojrzałych narzędzi, które nie obniżą jakości pracy. Pracownicy nie powinni być zmuszani do używania niedojrzałych, przypadkowych aplikacji tylko dlatego, że są otwarte. Konkurs powinien promować rozwiązania stabilne, z dobrą dokumentacją, aktywną społecznością, wsparciem profesjonalnym i realnymi wdrożeniami. Dobre doświadczenie użytkownika jest warunkiem akceptacji. Open source ma być profesjonalną alternatywą, a nie symbolem obniżenia standardu pracy.

Czternastym warunkiem jest pokazywanie sukcesów z obszarów, w których open source już działa. Szczególnie dobrym przykładem jest GIS. QGIS jest profesjonalnym narzędziem używanym realnie jako alternatywa dla komercyjnych systemów GIS, a PostgreSQL/PostGIS daje administracji otwartą, skalowalną bazę danych przestrzennych. W wielu jednostkach pracownicy geodezji, planowania przestrzennego, ochrony środowiska, transportu czy zarządzania kryzysowego już widzą, że otwarte oprogramowanie może być narzędziem wysokiej jakości. Warto wykorzystywać te przykłady w komunikacji: open source to nie eksperyment, ale sprawdzony model pracy w profesjonalnych dziedzinach.

Piętnastym warunkiem jest współpraca ze społecznościami. Polska i europejska społeczność open source, w tym środowiska QGIS (Stowarzyszenie QGIS Polska), PostgreSQL, Linux, FOSS4G, OpenStreetMap i otwartych danych, dysponuje dużą wiedzą praktyczną. Warto włączać ją w szkolenia, webinary, konsultacje, tworzenie materiałów, testowanie narzędzi i budowę dobrych praktyk. Społeczności nie zastępują formalnego wsparcia, ale mogą znacząco zwiększyć jakość wdrożeń i pomóc przełamać stereotyp, że open source jest pozbawiony zaplecza.

Szesnastym warunkiem jest mierzenie adopcji. O powodzeniu migracji nie świadczy wyłącznie liczba zainstalowanych aplikacji. Należy mierzyć, ilu pracowników realnie korzysta z nowych narzędzi, jakie zadania wykonują, ile zgłoszeń wpływa do helpdesku, jakie problemy się powtarzają, które szkolenia są najpotrzebniejsze, ile dokumentów powstaje w otwartych formatach i gdzie nadal występuje zależność od rozwiązań własnościowych. Takie dane pozwolą poprawiać wdrożenie na bieżąco, zamiast oceniać je dopiero po zakończeniu pilotażu.

Siedemnastym warunkiem jest akceptacja krzywej uczenia. Nawet dobre narzędzie wymaga zmiany nawyków. Nie należy oczekiwać, że wszyscy użytkownicy natychmiast będą pracować tak samo szybko jak wcześniej. W harmonogramie pilotażu trzeba przewidzieć czas na naukę, korekty, dodatkowe szkolenia i poprawę szablonów. Przejściowy spadek komfortu części użytkowników nie powinien być traktowany jako porażka, jeżeli projekt ma właściwe wsparcie i prowadzi do trwałego uniezależnienia administracji.

Osiemnasty warunek to silne wsparcie kierownictwa. Pracownicy muszą widzieć, że migracja jest decyzją strategiczną, a nie pobocznym eksperymentem działu IT. Kierownictwo jednostki powinno jasno komunikować cele, zapewnić czas na szkolenia, wspierać lokalnych liderów, reagować na problemy i nie dopuszczać do równoległego sabotowania zmiany przez utrzymywanie starych praktyk bez ograniczeń. Bez wsparcia kierownictwa nawet najlepsze szkolenia nie wystarczą.

Podsumowując, najskuteczniejsze przejście pracowników z rozwiązań własnościowych na open source zapewni połączenie działań technicznych, organizacyjnych i edukacyjnych: audyt sposobów pracy, migracja etapowa, przygotowanie szablonów, szkolenia zadaniowe, materiały porównawcze, lokalni ambasadorzy, intensywny helpdesk po starcie, centralne centrum kompetencji, wsparcie administratorów, mierzenie adopcji i jasna komunikacja celu. Najważniejsze jest, aby pracownik nie czuł, że „odebrano mu znane narzędzie”, ale że otrzymał nowe, profesjonalne środowisko pracy wspierające niezależność państwa, bezpieczeństwo danych i racjonalne wydatkowanie pieniędzy publicznych.

6. Jakie wsparcie techniczne i eksperckie powinno zostać zapewnione podmiotom realizującym pilotaże?

Podmiotom realizującym pilotaże należy zapewnić wsparcie techniczne i eksperckie obejmujące cały cykl życia wdrożenia: od analizy przedwdrożeniowej, przez wybór architektury, migrację, szkolenia, bezpieczeństwo, uruchomienie produkcyjne, aż po utrzymanie i replikację w innych jednostkach. Wsparcie nie powinno ograniczać się do jednorazowej pomocy wykonawcy w instalacji systemu. Konkurs powinien budować trwałą zdolność administracji do samodzielnego wyboru, utrzymania, audytowania, rozwijania i zmiany technologii.

Wyniki badania MC-COI bardzo wyraźnie pokazują, że jednostki oczekują przede wszystkim centralnego wsparcia technicznego, jasnych wytycznych państwowych, katalogu zweryfikowanych rozwiązań open source, finansowania migracji i szkoleń oraz księgi dobrych praktyk. To powinno być potraktowane jako punkt wyjścia dla konstrukcji konkursu. Administracja, zwłaszcza mniejsze JST, nie powinna być pozostawiona sama z decyzjami technologicznymi, licencyjnymi, bezpieczeństwa i organizacyjnymi.

Najważniejszym elementem powinno być utworzenie rządowego centrum kompetencji open source dla administracji publicznej. Takie centrum mogłoby działać jako stałe zaplecze merytoryczne konkursu, a docelowo jako instytucjonalny filar polityki open source w państwie. Jego zadaniem nie powinno być zastępowanie rynku usług IT, lecz standaryzacja, doradztwo, audyt, wymiana doświadczeń, prowadzenie katalogu rozwiązań, wsparcie drugiej linii oraz zapewnienie, że efekty pilotaży będą replikowalne w innych jednostkach.

Centrum kompetencji powinno zapewniać co najmniej następujące rodzaje wsparcia:

1. wsparcie techniczne drugiej linii dla beneficjentów i lokalnych administratorów;
2. doradztwo architektoniczne przy projektowaniu środowiska pracy;
3. doradztwo licencyjne dotyczące wyboru, łączenia i publikacji komponentów open source;
4. doradztwo w zakresie zamówień publicznych;
5. wsparcie w obszarze cyberbezpieczeństwa i łańcucha dostaw oprogramowania;
6. katalog zweryfikowanych rozwiązań open source dla administracji;
7. wzorcowa dokumentacja, procedury i checklisty wdrożeniowe;
8. repozytorium kodu, konfiguracji, materiałów szkoleniowych i dobrych praktyk;
9. koordynację współpracy między jednostkami realizującymi pilotaże;
10. wsparcie w mierzeniu efektów, kosztów i poziomu adopcji.

Pierwszą kategorią wsparcia powinno być wsparcie przedwdrożeniowe. Przed rozpoczęciem migracji jednostka powinna otrzymać pomoc w inwentaryzacji obecnego środowiska: używanego oprogramowania, licencji, kosztów, integracji, formatów danych, kont użytkowników, uprawnień, szablonów dokumentów, makr, procedur pracy, systemów dziedzinowych i ryzyk bezpieczeństwa. Bez takiego audytu nie da się dobrze zaprojektować migracji ani ocenić, które elementy można zastąpić szybko, a które wymagają okresu przejściowego.

Drugą kategorią powinno być doradztwo architektoniczne. Jednostki potrzebują pomocy w wyborze modelu wdrożenia: lokalnie, w chmurze prywatnej, w modelu współdzielonym przez kilka JST, hybrydowo albo jako usługa zarządzana przez zaufanego operatora. Doradztwo powinno obejmować dobór komponentów, integrację z istniejącą infrastrukturą, SSO, pocztę, kalendarze, chmurę plików, pakiet biurowy, wideokonferencje, komunikację, backup, monitoring, zarządzanie tożsamością i dostępem oraz systemy dziedzinowe. Wsparcie powinno promować architekturę modułową, w której poszczególne komponenty są zintegrowane, ale pozostają wymienne.

Trzecią kategorią powinien być katalog zweryfikowanych rozwiązań open source. Nie chodzi o prostą listę programów, lecz o praktyczne narzędzie decyzyjne dla administracji. Katalog powinien zawierać informacje o licencji, dojrzałości projektu, aktywności społeczności, dostępności wsparcia, modelu aktualizacji, poziomie dokumentacji, ryzykach bezpieczeństwa, zgodności ze standardami, możliwościach integracji, wymaganiach infrastrukturalnych, przykładach wdrożeń i orientacyjnych kosztach utrzymania. Powinien obejmować m.in. pakiety biurowe, pocztę, kalendarze, komunikatory, wideokonferencje, chmurę plików, bazy danych, GIS, helpdesk, monitoring, bezpieczeństwo, zarządzanie tożsamością i narzędzia administracyjne.

Czwartą kategorią powinno być wsparcie licencyjne. Wiele jednostek nie ma wystarczających kompetencji do samodzielnej oceny licencji open source, ich kompatybilności, obowiązków publikacyjnych, zasad łączenia komponentów, konsekwencji copyleft, open core czy modeli SaaS. Centrum kompetencji powinno przygotować wzorcowe klauzule, przewodniki, listę akceptowalnych licencji, rekomendacje dotyczące EUPL 1.2 dla rezultatów finansowanych publicznie oraz procedurę publikowania kodu i dokumentacji. Wsparcie licencyjne powinno ograniczać ryzyko, że jednostki wybiorą rozwiązania pozornie otwarte lub trudne do ponownego użycia.

Piątą kategorią powinno być wsparcie w zamówieniach publicznych. Jednostki potrzebują wzorcowych zapisów do SWZ i umów, które zabezpieczą interes publiczny: dostęp do kodu źródłowego, przekazanie dokumentacji, otwarte formaty, przenaszalność danych, brak zamkniętych komponentów krytycznych, możliwość zmiany wykonawcy, plan wyjścia, wymagania bezpieczeństwa, SLA, obowiązek publikacji rezultatów i prawa do ponownego użycia. Bez takich wzorców wiele urzędów może nieświadomie odtworzyć vendor lock-in, tylko w nowej formule.

Szóstą kategorią powinno być wsparcie bezpieczeństwa. Każdy pilotaż powinien otrzymać pomoc w przygotowaniu analizy ryzyka, modelu uprawnień, procedury aktualizacji, SBOM, monitorowania podatności, backupu, odtwarzania, logowania, hardeningu, zarządzania sekretami i reagowania na incydenty. Wsparcie powinno obejmować również ocenę łańcucha dostaw oprogramowania, w tym pochodzenia pakietów, obrazów kontenerów i zależności. Open source daje możliwość audytu i przejrzystości, ale wymaga dojrzałych procedur utrzymaniowych. Szczególnie mniejsze JST powinny mieć dostęp do ekspertów bezpieczeństwa, których samodzielne zatrudnienie byłoby dla nich trudne.

Siódmą kategorią powinien być helpdesk i wsparcie operacyjne. W projekcie powinien istnieć jasny podział na pierwszą, drugą i trzecią linię wsparcia. Pierwsza linia może działać lokalnie w jednostce lub u wykonawcy. Druga linia powinna być dostępna

centralnie dla administratorów i koordynatorów pilotażu. Trzecia linia powinna obejmować ekspertów od konkretnych komponentów, bezpieczeństwa, baz danych, poczty, chmury, dokumentów, GIS czy integracji. Ważne jest, aby wsparcie było dostępne szczególnie intensywnie w pierwszych tygodniach po uruchomieniu produkcyjnym.

Ósmą kategorią powinno być wsparcie szkoleniowe. Konkurs powinien finansować nie tylko szkolenia użytkowników końcowych, ale także szkolenia osób zajmujących się zamówieniami publicznymi (w zakresie otwartych standardów), administratorów, kierowników projektów, osób odpowiedzialnych za zamówienia publiczne, inspektorów ochrony danych, osób odpowiedzialnych za bezpieczeństwo i lokalnych ambasadorów zmiany. Szkolenia powinny być zadaniowe i praktyczne: jak obsłużyć dokument, jak pracować z kalendarzem, jak udostępnić plik, jak zorganizować wideokonferencję, jak wykonać backup, jak nadać uprawnienia, jak zareagować na incydent, jak opublikować dane przestrzenne w standardzie otwartym. Potrzebne są też krótkie materiały referencyjne, e-learning, webinary, nagrania, podręczniki i forum pytań.

Dziewiątą kategorią powinno być wsparcie migracyjne. Jednostki potrzebują ekspertów od migracji dokumentów, poczty, kalendarzy, kontaktów, kont użytkowników, uprawnień, szablonów, makr, danych przestrzennych i baz danych. Migracja nie może polegać na prostym skopiowaniu plików. W wielu przypadkach konieczne będzie oczyszczenie danych, uporządkowanie katalogów, konwersja formatów, przegląd szablonów, identyfikacja krytycznych arkuszy kalkulacyjnych, testy kompatybilności oraz przygotowanie procedur dla dokumentów przychodzących i wychodzących. Wsparcie migracyjne powinno zmniejszać ryzyko przestoju i frustracji użytkowników.

Dziesiątą kategorią powinno być wsparcie dotyczące otwartych standardów i interoperacyjności. Podmioty realizujące pilotaże powinny otrzymać wytyczne, kiedy i jak stosować ODF, PDF/A, CSV, XML, JSON, OpenAPI, standardy OGC, GeoPackage, WMS, WFS, OGC API i inne otwarte standardy. Należy jasno rozróżnić obsługę formatów dominujących na rynku w okresie przejściowym od docelowego budowania suwerenności na formatach otwartych. W przeciwnym razie administracja może wymienić aplikację, ale nadal pozostać zależna od zamkniętych formatów i niejawnych sposobów wymiany danych.

Warto tu wspomnieć także o potrzebie regulacji formatów danych opisanych w Rozporządzeniu Rady Ministrów z dnia 21 maja 2024 r. w sprawie *Krajowych Ram Interoperacyjności, minimalnych wymagań dla rejestrów publicznych i wymiany informacji w postaci elektronicznej oraz minimalnych wymagań dla systemów teleinformatycznych* wygaszonym z dniem 23 lutego 2027 roku zgodnie z Ustawą z dnia 25 lipca 2025 r. o *zmianie ustawy o informatyzacji działalności podmiotów realizujących zadania publiczne oraz niektórych innych ustaw*. Nowe rozporządzenie powinno przewidywać kwestie własnościowych i zamkniętych formatów danych w administracji.

Jedenastą kategorią powinno być wsparcie w zakresie replikowalności. Każdy pilotaż powinien otrzymać pomoc w przygotowaniu pakietu replikacyjnego: dokumentacji architektury, instrukcji instalacji, skryptów, plików konfiguracyjnych, materiałów szkoleniowych, opisów integracji, listy problemów, procedur aktualizacji, procedur backupu i planu wyjścia. Centrum kompetencji powinno sprawdzać, czy pakiet jest zrozumiały i

użyteczny dla innej jednostki. Można rozważyć obowiązek testowego odtworzenia wdrożenia w środowisku referencyjnym.

Dwunastą kategorią powinno być wsparcie finansowo-organizacyjne. Open source nie oznacza braku kosztów. Koszty przesuwają się z licencji i subskrypcji na wdrożenie, migrację, utrzymanie, bezpieczeństwo, dokumentację, szkolenia i wsparcie. Konkurs powinien dopuszczać finansowanie tych elementów, ponieważ to one decydują o sukcesie. Szczególnie ważne jest zapewnienie finansowania wsparcia po uruchomieniu systemu, a nie tylko do dnia odbioru technicznego.

Trzynastą kategorią powinno być wsparcie trwałości po pilotażu. Już we wniosku jednostka powinna przedstawić plan utrzymania na co najmniej dwa–trzy lata po zakończeniu finansowania pilotażowego. Plan powinien obejmować źródła finansowania, odpowiedzialność za aktualizacje, szkolenia nowych pracowników, sposób obsługi zgłoszeń, bezpieczeństwo, backup, umowy wsparcia i udział w repozytorium wspólnych rozwiązań. Bez takiego planu projekt może zakończyć się sukcesem formalnym, ale nie przetrwać organizacyjnie.

Czternastą kategorią powinno być wsparcie w wyborze profesjonalnych usług wsparcia dla krytycznych komponentów open source. W przypadku najważniejszych elementów infrastruktury, takich jak system operacyjny serwerów, chmura prywatna, poczta, baza danych, system tożsamości, backup czy platforma dokumentów, warto przewidywać finansowanie oficjalnego wsparcia producenta, fundacji, integratora lub wyspecjalizowanej firmy. Jest to nadal zgodne z ideą open source: płacimy nie za zamknięcie kodu, lecz za kompetencje, SLA, aktualizacje, odpowiedzialność i bezpieczeństwo. Taki model jest często korzystniejszy niż ponoszenie stałych kosztów licencji bez realnej kontroli nad technologią.

Piętnastą kategorią powinno być wsparcie eksperckie dziedzinowe. Nie wszystkie wdrożenia są takie same. Innych kompetencji wymaga pakiet biurowy, innych poczta i kalendarz, innych EZD, innych GIS, innych baza danych, a innych bezpieczeństwo. W obszarze GIS warto szczególnie wykorzystać doświadczenie społeczności QGIS (Stowarzyszenie QGIS Polska), PostgreSQL/PostGIS, FOSS4G i otwartych standardów OGC. QGIS oraz PostgreSQL/PostGIS pokazują, że profesjonalne open source może być realną alternatywą dla komercyjnych rozwiązań klasy ArcGIS Enterprise i Oracle Spatial, pod warunkiem właściwego wdrożenia, szkolenia i utrzymania. Podobne wsparcie dziedzinowe powinno powstać dla innych klas systemów.

Szesnastą kategorią powinno być wsparcie społeczności praktyków. Jednostki realizujące pilotaże powinny mieć możliwość regularnej wymiany doświadczeń: spotkań online, warsztatów, repozytorium pytań i odpowiedzi, list dyskusyjnych, konferencji technicznych i grup roboczych. Jednostki, które już wdrożyły określone rozwiązanie, powinny wspierać jednostki rozpoczynające wdrożenie. Taki model mentorstwa między urzędami zwiększa zaufanie, skraca czas wdrożeń i ogranicza zależność od jednego integratora.

Siedemnastą kategorią powinno być wsparcie ewaluacyjne. Podmioty realizujące pilotaże powinny otrzymać wspólną metodykę mierzenia efektów: kosztów, poziomu adopcji, liczby użytkowników, liczby zgłoszeń, czasu reakcji, liczby dokumentów tworzonych w otwartych formatach, jakości szkoleń, stabilności systemu, liczby incydentów, stopnia

replikowalności i satysfakcji użytkowników. Bez wspólnej metodyki trudno będzie porównać pilotaże i wyciągnąć wnioski dla kolejnych edycji programu.

Osiemnastą kategorią powinno być wsparcie komunikacyjne. Przejście na open source powinno być przedstawiane pracownikom i kierownictwu jako element modernizacji państwa, suwerenności cyfrowej, bezpieczeństwa danych i racjonalnego wydatkowania środków publicznych. Potrzebne są gotowe materiały komunikacyjne, prezentacje, FAQ, argumentarium dla kierownictwa i wzory informacji dla pracowników. Zmiana będzie łatwiejsza, jeśli jej sens zostanie dobrze wyjaśniony.

Podsumowując, podmiotom realizującym pilotaże należy zapewnić wsparcie wielowarstwowe: centralne centrum kompetencji, helpdesk drugiej linii, katalog zweryfikowanych rozwiązań, wytyczne państwowe, doradztwo architektoniczne, licencyjne, prawne i zakupowe, wsparcie bezpieczeństwa, wsparcie migracyjne, szkolenia użytkowników i administratorów, repozytorium dobrych praktyk, pakiety replikacyjne, wsparcie dziedzinowe, finansowanie utrzymania oraz społeczność praktyków. Celem powinno być nie tylko sprawne uruchomienie kilku pilotaży, ale stworzenie trwałego ekosystemu, dzięki któremu kolejne jednostki administracji będą mogły wdrażać open source szybciej, bezpieczniej i taniej, bez ponownego popełniania tych samych błędów.

7. Pozostałe uwagi dotyczące planowanego konkursu

Planowany konkurs jest bardzo potrzebnym i pozytywnym krokiem w stronę cyfrowej suwerenności administracji publicznej. Warto jednak podkreślić, że jego znaczenie wykracza poza samo wdrożenie kilku narzędzi open source w wybranych jednostkach. Konkurs może stać się początkiem trwałej zmiany sposobu myślenia o technologiach publicznych: od modelu biernego kupowania licencji i subskrypcji do modelu świadomego zarządzania kodem, danymi, standardami, bezpieczeństwem i kompetencjami.

Pierwszą dodatkową uwagą jest konieczność prowadzenia szerokich działań świadomościowych, szczególnie na poziomie samorządowym. W praktyce dla dużej części użytkowników i części decydentów „komputer” nadal oznacza: Windows + Microsoft Office + ewentualnie Microsoft Teams/Outlook. Nie jest to zarzut wobec pracowników administracji, lecz naturalny efekt wieloletniej dominacji jednego modelu technologicznego. W badaniu MC-COI bardzo wyraźnie widać skalę tej dominacji: systemy operacyjne i pakiety biurowe są niemal całkowicie zdominowane przez rozwiązania własnościowe. Dlatego konkurs powinien być połączony z kampanią edukacyjną pokazującą, że komputer, dokument, arkusz, poczta, mapa czy baza danych nie muszą być utożsamiane z jednym producentem i jednym ekosystemem.

Ta zmiana świadomości jest równie ważna jak samo finansowanie wdrożeń. Bez niej nawet najlepsze technicznie pilotaże mogą spotkać się z oporem użytkowników, obawą kierownictwa lub nieufnością lokalnych działów IT. Pracownicy powinni rozumieć, że open source nie oznacza „gorszej darmowej kopii”, lecz inny model kontroli nad technologią: możliwość audytu, niezależność od jednego dostawcy, przenoszalność danych, większą konkurencyjność rynku wsparcia i lepsze wykorzystanie środków publicznych. Szczególnie

ważne jest wyjaśnianie różnicy między darmowym programem, programem otwartoźródłowym, otwartym standardem i otwartym formatem danych.

Drugą uwagą jest konieczność mocnego powiązania open source z otwartymi standardami. Sama wymiana aplikacji nie wystarczy, jeżeli dokumenty, dane, integracje i procedury nadal pozostaną zamknięte w formatach kontrolowanych przez jednego dostawcę. Administracja nie powinna przejść z jednego vendor lock-in do drugiego. Celem konkursu powinno być odzyskanie kontroli nie tylko nad kodem, ale także nad danymi i ich długoterminową czytelnością. Dlatego należy premiować ODF jako docelowy format dokumentów edytowalnych, PDF/A jako format archiwizacyjny, otwarte API, jawne schematy danych oraz otwarte standardy branżowe, np. standardy OGC w geoinformacji.

Trzecią uwagą jest potrzeba pragmatycznego podejścia do okresu przejściowego. Nie da się z dnia na dzień odciąć administracji od formatów i systemów, które przez lata były dominujące. Dlatego potrzebna jest interoperacyjność z formatami DOCX, XLSX, PPTX i istniejącymi systemami komercyjnymi. Nie może ona jednak stać się pretekstem do utrwalenia obecnej zależności. Konkurs powinien jasno odróżnić „obsługę formatów dominujących na rynku w okresie przejściowym” od „docelowego standardu pracy administracji”. Docelowo państwo powinno konsekwentnie wzmacniać otwarte formaty, bo tylko one zapewniają długoterminową niezależność i bezpieczeństwo archiwów publicznych.

Czwartą uwagą jest konieczność zmiany sposobu liczenia kosztów. Open source nie jest bezkosztowy, ale pozwala zmienić strukturę wydatków publicznych. Dziś znaczna część środków jest przeznaczana na licencje, subskrypcje i przymusowe aktualizacje rozwiązań własnościowych, często trafiające do globalnych korporacji technologicznych. W modelu open source środki można w większym stopniu kierować na lokalne i europejskie kompetencje: analizę, migrację, bezpieczeństwo, szkolenia, dokumentację, utrzymanie, rozwój funkcji i wsparcie użytkowników. Jest to korzystniejsze z punktu widzenia podatnika, rynku krajowego i odporności państwa.

Piątą uwagą jest potrzeba utworzenia trwałego centrum kompetencji open source dla administracji. Konkurs nie powinien być jednorazowym grantem, po którym jednostki zostają same z utrzymaniem systemów. Potrzebne jest miejsce, które będzie prowadzić katalog zweryfikowanych rozwiązań, wzorcowe dokumentacje, repozytorium rezultatów, wytyczne bezpieczeństwa, wzory zapisów do SWZ, materiały szkoleniowe, dobre praktyki i wsparcie drugiej linii dla JST. Takie centrum powinno pomagać jednostkom przed wdrożeniem, w trakcie wdrożenia i po jego zakończeniu.

Szóstą uwagą jest konieczność zapobiegania pozornej otwartości. Konkurs powinien uważać na rozwiązania typu open core, source available, quasi-open-source albo SaaS deklarowany jako open source, ale bez realnej przenaszalności danych i konfiguracji. Jeżeli krytyczne funkcje bezpieczeństwa, audytu, SSO, wysokiej dostępności, archiwizacji lub integracji są dostępne wyłącznie w zamkniętych modułach, to administracja ryzykuje nowe uzależnienie. Celem powinno być realne prawo do uruchamiania, audytowania, modyfikowania, utrzymywania i przenoszenia systemu.

Siódmą uwagą jest potrzeba bardzo mocnego kryterium replikowalności. Każde wdrożenie finansowane z konkursu powinno zostawić po sobie pakiet możliwy do użycia przez inne jednostki: kod, konfigurację, dokumentację, instrukcje, materiały szkoleniowe,

opis integracji, listę problemów, procedury bezpieczeństwa, plan utrzymania i plan wyjścia. Jeżeli rezultat pilotażu da się utrzymać tylko przez jednego wykonawcę i tylko w jednej jednostce, to nie jest to wdrożenie systemowo wartościowe. Państwo powinno płacić za rozwiązania, które zmniejszają koszt kolejnych wdrożeń.

Ósmą uwagą jest potrzeba przygotowania wzorcowych zapisów do zamówień publicznych. Wiele JST nie ma wyspecjalizowanych zespołów prawnych i technicznych, które potrafią samodzielnie opisać wymagania dotyczące open source, licencji, danych, API, dokumentacji, bezpieczeństwa i planu wyjścia. Bez gotowych klauzul istnieje ryzyko, że nawet dobrze pomyślany konkurs doprowadzi do lokalnych umów odtwarzających vendor lock-in. Ministerstwo powinno przygotować wzorcowe zapisy do SWZ i umów, które beneficjenci będą mogli bezpiecznie wykorzystać.

Dziewiątą uwagą jest konieczność włączenia środowisk eksperckich i społecznościowych. Polska ma znaczący kapitał kompetencyjny w społecznościach open source: Linux, PostgreSQL, QGIS, FOSS4G, OpenStreetMap, wolne standardy, cyberbezpieczeństwo, otwarte dane. Te środowiska nie powinny być traktowane jako margines lub wyłącznie jako hobbystyczna aktywność. Często skupiają ekspertów z wieloletnim doświadczeniem wdrożeniowym, akademickim i administracyjnym. Warto włączyć je w konsultacje, tworzenie katalogu rozwiązań, recenzowanie dokumentacji, szkolenia, webinary i budowę dobrych praktyk.

Dziesiątą uwagą jest szczególne wykorzystanie GIS jako przykładu sukcesu open source. Ekosystem QGIS + PostgreSQL/PostGIS + GDAL/OGR + GeoServer/MapServer + standardy OGC jest jednym z najlepszych dowodów, że open source może być profesjonalnym, dojrzałym i realnie używanym narzędziem administracji. QGIS jest alternatywą dla komercyjnych systemów klasy ArcGIS, a PostgreSQL/PostGIS może w wielu zastosowaniach zastępować kosztowne rozwiązania bazodanowe klasy Oracle Spatial. Co ważne, siła tego ekosystemu wynika nie tylko z braku opłat licencyjnych, ale z otwartych formatów, otwartych standardów, wielu dostawców wsparcia i aktywnej międzynarodowej społeczności FOSS4G, w której Polacy są istotnym elementem rozwoju.

Jedenastą uwagą jest potrzeba jasnego rozdzielenia trzech poziomów zmiany: aplikacji, danych i kompetencji. Wymiana aplikacji bez zmiany formatu danych daje tylko częściową niezależność. Zmiana formatu bez przeszkolenia użytkowników powoduje frustrację. Szkolenia bez stabilnej architektury technicznej nie przyniosą trwałego efektu. Konkurs powinien premiować projekty, które łączą te trzy warstwy: wdrożenie narzędzia, migrację do otwartych standardów i zbudowanie kompetencji po stronie urzędu.

Dwunastą uwagą jest potrzeba mierzenia efektów w sposób szerszy niż „liczba zainstalowanych stanowisk”. Wskaźniki powinny obejmować m.in.: liczbę realnych użytkowników, poziom wykorzystania nowych narzędzi, liczbę dokumentów w otwartych formatach, zmniejszenie kosztów licencyjnych, liczbę przeszkolonych osób, czas reakcji helpdesku, liczbę problemów rozwiązanych w bazie wiedzy, możliwość odtworzenia wdrożenia w innej jednostce, stopień przenaszalności danych oraz liczbę komponentów opublikowanych do ponownego użycia.

Trzynastą uwagą jest potrzeba uwzględnienia różnic między jednostkami. Inaczej wygląda sytuacja dużego miasta, inaczej małej gminy, a inaczej jednostki centralnej. Wiele

samorządów ma ograniczone kadry IT i nie może samodzielnie utrzymywać skomplikowanej infrastruktury. Dlatego konkurs powinien przewidywać różne modele referencyjne: dla małych JST, średnich JST, dużych miast i jednostek centralnych. W niektórych przypadkach najlepszy może być model współdzielonej infrastruktury, centrum usług wspólnych albo zarządzanej chmury prywatnej, pod warunkiem zachowania kontroli nad danymi i możliwości zmiany operatora.

Czternastą uwagą jest konieczność zachowania realizmu i entuzjazmu jednocześnie. Open source nie rozwiąże wszystkich problemów administracji automatycznie. Wymaga kompetencji, utrzymania, dokumentacji, szkoleń i odpowiedzialności. Jednocześnie jest jedną z najlepszych dróg do uniezależnienia Europy od technologicznej dominacji USA i Chin, ograniczenia kosztów publicznych oraz odbudowy lokalnych kompetencji. Konkurs powinien być komunikowany właśnie w ten sposób: jako odpowiedzialna, profesjonalna i europejska modernizacja państwa, a nie jako eksperyment ideologiczny.

Piętnastą uwagą jest potrzeba zapewnienia ciągłości po zakończeniu pilotażu. Projekty powinny zawierać plan utrzymania na kolejne lata: kto aktualizuje system, kto zapewnia wsparcie, kto reaguje na incydenty, kto prowadzi szkolenia dla nowych pracowników, kto utrzymuje dokumentację i kto odpowiada za replikację rozwiązania. Bez takiego planu nawet udany pilotaż może po roku zostać uznany za trudny w utrzymaniu i wyparty przez powrót do rozwiązań własnościowych.

Szesnąstą uwagą jest promocja pozytywnych przykładów. W administracji trzeba pokazywać konkretne historie sukcesu: QGIS w geodezji i planowaniu przestrzennym, PostgreSQL/PostGIS w zarządzaniu danymi, LibreOffice i ODF w dokumentach, Nextcloud w chmurze plików, rozwiązania open source w cyberbezpieczeństwie, monitoringu i automatyzacji. Takie przykłady są skuteczniejsze niż abstrakcyjne deklaracje. Pracownik samorządu musi zobaczyć, że inni już tego używają i że to działa.

Podsumowując, najważniejszą "pozostałą uwagą" jest to, że konkurs powinien być nie tylko programem wdrożeniowym, ale także programem zmiany świadomości. Trzeba przełamać głęboko zakorzenione przekonanie, że komputer równa się Windows, dokument równa się DOCX, arkusz równa się Excel, a praca biurowa równa się Microsoft Office. Administracja publiczna musi odzyskać wyobraźnię technologiczną i zdolność wyboru. Open source oraz otwarte standardy dają taką możliwość, ale wymagają wsparcia, edukacji, cierpliwości i konsekwentnej polityki państwa. Konkurs może być bardzo ważnym pierwszym krokiem — pod warunkiem, że zostanie zaprojektowany jako początek trwałej transformacji, a nie jednorazowy pilotaż kilku aplikacji.

W imieniu Stowarzyszenia QGIS Polska zespół redagujący:

[Redacted signature block]

[Redacted signature block]